Umedicine: A System for Clinical Practice Support and Data Analysis

Nuno F. Lages[1], Bernardo Caetano[1], Manuel J. Fonseca[2], João D. Pereira[1], Helena Galhardas[1], and Rui Farinha[3]

[1] INESC-ID and Instituto Superior Técnico, Universidade de Lisboa, Portugal
[2] LaSIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
[3] Hospital de São José, Portugal

Abstract. Recording patient's clinical information in a comprehensive and easy way is very important for health care providers. However, and although there are information systems to facilitate the storage and access to patient's data, many records are still in paper. This happens because systems often are complex to use and do not provide means to gather statistical information about a population of patients, thus limiting the usefulness of the data. Physicians often give up searching for relevant information to support their medical decisions because the task is too time-consuming. This paper proposes Umedicine, a web-based software application in Portuguese that addresses current limitations of clinical information systems. Umedicine is an application for physicians, patients and administrative staff that keeps, in a structured way, medical information (e.g., symptoms, clinical examination results, and treatments prescribed) up to date on a platform that provides easy and quick access to a large amount of clinical data. Furthermore, the paper presents the application of a particular clustering algorithm and a visualization module for analyzing patient time series data, to identify evolution patterns. Preliminary user tests revealed promising results, showing that users were able to identify the evolution of groups of patients over time and their common characteristics.

Keywords: clinical information system, data analysis, clustering, data visualization

1 Introduction

For health care providers, recording every patient's clinical information comprehensively is of paramount importance. However, for many years, this information has been recorded in paper, and kept in large dedicated archives, making it difficult to retrieve relevant past clinical information quickly and effectively when required for patient care or for scientific research purposes.

With the widespread use of computers, new software tools were developed for clinical staff to record information about patients. This easy access to the clinical data promises a significant impact in clinical practice. In particular, continuous patient monitoring can be ubiquitous, enabling fast response by clinical staff and quick situation assessment by physicians; clinical research can benefit from much larger, easier-to-access data sources, that will accelerate the discovery of new medical knowledge; and health care managers can be able to make more informed decisions regarding institutional policies.

Currently, existing medical information systems are often too complex to use and provide free-form fields for collecting patient's medical information, which therefore is not stored as structured data. For this reason, many patient records are still in paper and, in the cases where the information is in the system, it is not possible to gather statistical information about a population of patients, thus limiting the usefulness of the data. As a consequence, physicians often give up searching for relevant information to support medical decisions because the task is too time-consuming [2, 10].

Hence, there still is a need of tools to enable integration and analysis of clinical data in an effective manner [10, 11] and to make it useful in everyday practice. Physicians should have means to explore existing patient data quickly to support their diagnosis, treatment decisions and research.

In this paper, we propose *Umedicine*, a new medical information system that addresses the current limitations of clinical information systems. Umedicine is a web-based application for physicians, patients and administrative staff with an appealing and easy-to-use user interface. Through it, physicians and patients keep medical information up to date on a platform that provides easy, quick and always-on access to a large amount of clinical data. The medical information is persistently kept in an electronic and structured format, thus enabling the visualization of medical data per patient as well as the application of data analysis techniques to extract interesting knowledge from the collected medical data for a given population of patients. To the best of our knowledge, our application is the only one that supports: (i) filling in standard international diagnosis-support questionnaires, (ii) collecting and storing patient data (personal data, and clinical data such as symptoms, clinical examination results, identified pathologies, and treatments prescribed) in a structured way, appropriate to each medical specialty, and (iii) applying data analysis algorithms, all integrated in one software platform. In particular, we demonstrate the use of a clustering algorithm and a visualization mechanism for analysis of patient time series data.

We performed a preliminary evaluation of the Umedicine user interface with users, asking them to perform several task scenarios. Users also filled in a satisfaction questionnaire to qualitatively assess Umedicine. Achieved results are promising, as users were able to complete with success all the tasks and were mostly satisfied with the user interface provided.

2 Architecture and Implementation

Umedicine has a client-server style architecture with three main components (see Fig. 1): (i) a front-end available to the user through a web browser client, (ii) a back-end server, and (iii) a relational database. Users interact with the system through a web browser, which submits HTTP requests to the server. The

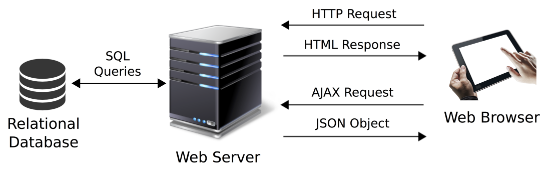


Fig. 1. Architecture of the Umedicine system.

server returns an HTML page. Data may be asynchronously requested from the server through Asynchronous JavaScript and XML (Ajax) requests made by the browser. To respond to Ajax requests, the server queries a relational database and returns the data to the client in the form of JSON objects. The use of Ajax allows to exchange only new necessary data to fill the page, minimizing data transfers from the server and improving user waiting time. The server is developed in Java with extensive use of Spring Framework[4]. Web pages are generated with the JavaServer Pages (JSP) and Apache Tiles[5] technologies.

Umedicine supports four types of users: non-administrator physicians, administrator physicians, clerks and patients. Non-administrator physicians can add new patients to the system and search, read and modify patient personal and medical information. In addition to having access to all the functionalities available to non-administrator physicians, administrator physicians can add new clerks and physicians (both administrator and non-administrator) to the system. Patients can view and modify their own personal information, view their own examination results and history and fill medical questionnaires for diagnosis support. Clerks can add new patients to the system and add limited patient personal information (name, birth date, contact and profession). The application is implemented in a way that ensures that each type of user can only use the functionalities and have access to information as described above. Data confidentiality is guaranteed in communications between clients and server by use of Transport Layer Security (TLS) encryption.

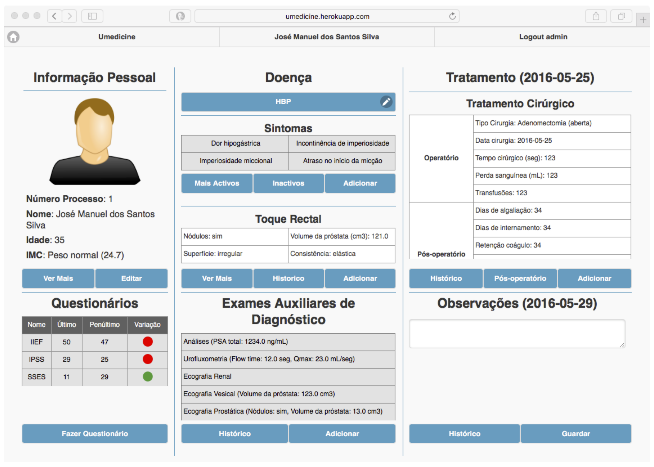
2.1 Front-end

Umedicine is implemented as a web application, providing responsive user interfaces for the different types of user. This way, Umedicine can be used both on mobile and desktop environments. In this section, we focus on the user interfaces offered for physicians and patients.

When the physician selects a patient to visualize, his/her information page is shown, as illustrated in Fig. 2. It is organized in six parts:

– **Personal information** (*Informação Pessoal*): area where physicians can view and modify information such as name, birth date, contact and habits of the patient;

[4] https://spring.io/
[5] https://tiles.apache.org/



Número Processo: 1
Nome: José Manuel dos Santos Silva
Idade: 35
IMC: Peso normal (24.7)

Dor hipogástrica	Incontinência de imperiosidade
Imperiosidade miccional	Atraso no início da micção

Nódulos: sim	Volume da próstata (cm3): 121.0
Superfície: irregular	Consistência: elástica

Nome	Último	Penúltimo	Variação
IIEF	50	47	🔴
IPSS	29	25	🔴
SSES	11	29	🟢

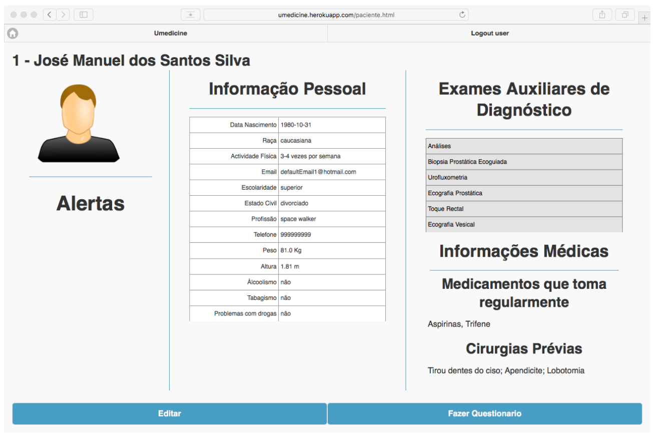
Análises (PSA total: 1234.0 ng/mL)
Urofluxometria (Flow time: 12.0 seg, Qmax: 23.0 mL/seg)
Ecografia Renal
Ecografia Vesical (Volume da próstata: 123.0 cm3)
Ecografia Prostática (Nódulos: sim, Volume da próstata: 13.0 cm3)

Operatório	Tipo Cirurgia: Adenomectomia (aberta)
	Data cirurgia: 2016-05-25
	Tempo cirúrgico (seg): 123
	Perda sanguínea (mL): 123
	Transfusões: 123
Pós-operatório	Dias de algaliação: 34
	Dias de internamento: 34
	Retenção coágulo: 34

Fig. 2. A patient's information page as viewed by a physician using Umedicine.

– **Disease** (*Doença*): area where physicians can select the medical condition that applies to the selected patient and view or modify information about symptoms; following the development team's physician advice, this area also includes the rectal examination information; users can view symptom and rectal examination histories and add new symptoms and examination results;
– **Treatment** (*Tratamento*): this area displays information about ongoing treatments and provides means to view treatment history and add new treatments;
– **Questionnaires** (*Questionários*): this area provides access to three diagnosis support and internationally accepted questionnaires that a patient can answer at home or during a medical appointment; these questionnaires are used to compute scores that represent the severity of a patient's condition; Umedicine currently supports three relevant standard questionnaire-based scores: State Self Esteem Scale (IIEF), International Prostate Symptom Score (IPSS) and State Self Esteem Scale (SSES);
– **Diagnosis support examinations** (*Exames Auxiliares de Diagnóstico*): this area shows the most recent results for several kinds of medical examinations and laboratory tests; the user can add new results and access the patient's examination and test histories;
– **Notes** (*Observações*): physicians can write a textual note about the patient's condition in the day of the medical appointment and view notes from previous appointments.

The user interface available to patients is more limited (Fig. 3) containing an area on the left where they can view their current photo and alerts requiring attention. The middle area shows their current personal information. On the right, the patient can view information about examinations, current medication and past surgeries. At the bottom, patients can choose to edit their personal



1 - José Manuel dos Santos Silva

Alertas

Informação Pessoal

Data Nascimento	1980-10-31
Raça	caucasiana
Actividade Física	3-4 vezes por semana
Email	defaultEmail1@hotmail.com
Escolaridade	superior
Estado Civil	divorciado
Profissão	space walker
Telefone	999999999
Peso	81.0 Kg
Altura	1.81 m
Álcoolismo	não
Tabagismo	não
Problemas com drogas	não

Exames Auxiliares de Diagnóstico

- Análises
- Biopsia Prostática Ecoguiada
- Urofluxometria
- Ecografia Prostática
- Toque Rectal
- Ecografia Vesical

Informações Médicas

Medicamentos que toma regularmente

Aspirinas, Trifene

Cirurgias Prévias

Tirou dentes do ciso; Apendicite; Lobotomia

Fig. 3. A patient's information page as viewed by the patient using Umedicine.

information or to answer a diagnosis support questionnaire. Note that this can be performed away from the clinic, which saves time during the medical appointment and give more privacy to the patient.

2.2 Back-end Server

The back-end server replies to the requests sent by the browsers, performs operations on the data and stores or retrieves data as necessary from the database. It is responsible for ensuring that each user has access to, and only to, the data he/she needs, and for loading the data or web pages requested by the users. The server is implemented as a Java application with extensive use of the Spring Framework ecosystem[6]. It follows the Model-View-Controler (MVC) design pattern and is composed of three main layers — from top to bottom: controllers, services and Data Access Objects (DAOs) (Fig. 4).

Controllers are the components responsible for the interaction with the clients. They receive client requests, invoke the appropriate business logic methods and send a response if required. Controller code (as well as service and Data Access Object (DAO) code) is, as much as possible, organized to maximize separation of concerns. In other words, a controller is written to support a specific set of coherent functionalities. For example, the code responsible for handling requests related to user account creation is supplied by methods of the class *UserController*; requests from pages where users fill questionnaires are handled by the *QuestionnaireController* class; and so on.

Any business logic-related processing of the data retrieved from the database or sent by the clients takes place in the *Services* layer. Service methods are invoked by controllers (or other services) and return data that controllers use to

[6] https://spring.io

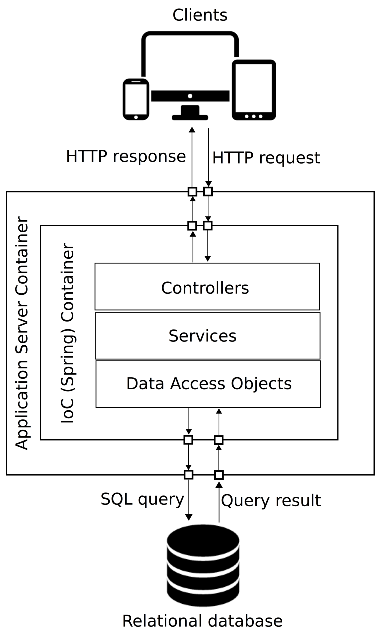


Fig. 4. Umedicine's server component architecture.

respond to the requests issued by the client. Service code is also organized according to functionalities. *UserService*, for instance, contains the code responsible for user account creation, performing duties such as invoking the DAO methods that store user data in the database (after performing any required data processing) or sending emails with new passwords to new users. In a similar fashion, there is a *QuestionnaireService* class that handles questionnaire data, and so on. Services also organize data retrieved from the database via DAOs into objects suitable to be used by the controllers, which may return data to the browser.

The bottom layer, DAOs, contains the code that enables the interaction between the server and the database. This code includes the SQL queries that insert new rows, update existing rows and retrieve data from the database tables. Information retrieved from the database is converted into Java objects and passed on to the upper service layer for further processing.

2.3 Relational Database

The Umedicine architecture encloses a relational database to persistently store three kinds of data: (i) clinical data inserted by patients or physicians; (ii) user authentication data; and (iii) metadata concerning a specific medical field. The current version of the application has a database that is set up for Urology but its schema is designed to be adaptable to other medical specialties. For this end, the metadata that corresponds to domain knowledge – drug names, disease names, symptoms, medical examinations and diagnosis support questionnaires

– is stored in the database rather than in the back-end server and is accessed when the application loads. With this approach, different domain knowledge from other medical specialties can be replaced into the database and plugged into the server with minimal modification of server and client code (ideally no modification will be needed whatsoever). This approach also enables switching easily from metadata in Portuguese to metadata in another language.

To illustrate the Umedicine database model, we present the subset of the database relational schema that models medical examinations. Primary keys of relations are underlined and foreign keys are specified by FK:

Clinical metadata:

ExaminationType (typeName, subTypeName)
Parameter (paramName, paramUnit, paramType)
ExaminationParameter (typeName, subTypeName, paramName)
  typeName, subTypeName: FK (ExaminationType)
  paramName: FK (Parameter)

Clinical data:

Patient (pNumber)
PerformedExamination (pExamID, pNumber, typeName, subTypeName, date)
  pNumber: FK (Patient)
  typeName, subTypeName: FK (ExaminationType)
PerformedExaminationValue (pExamID, paramName, value)
  pExamID: FK (PerformedExamination)
  paramName: FK (Parameter)

There are two types of data modeled in this relational schema: (i) clinical metadata and (ii) clinical data. ExaminationType, Parameter and Examination-Parameter are relations that model metadata. This metadata is accessed when the application starts and is used to determine the information that is shown in the user interface. The ExaminationType table stores all the possible types and subtypes of medical examinations. The tuples: <*Urofluxometry, -* > and <*Blood Tests, Liver function*> are examples of records stored in this table. Urofloxometry is a medical examination by itself while there are several kinds of blood tests so the subtype field is required. The Parameter table stores the existing parameters, their measurement units and expected types, i.e., String, Integer, Float or enumerated (in the case the type is ENUM, there is another table not represented here that stores the possible values). The tuples: <*Creatinine, mg/dl, float*> and <*Volume, ml, float*> are examples of records of the Parameter table. The table ExaminationParameter stores the correspondence between an examination and the its parameters. In accordance with the previous examples, the following tuples: <*Urofluxometry, -, Volume*> and <*Blood Tests, Liver function, Creatinine*> are examples of tuples of the ExaminationParameter table.

Volume (mL)

Fig. 5. Parameter Volume of type FLOAT and unit milliliter (mL)

The tables Patient, PerformedExamination and PerformedExaminationValue store clinical data collected through the usage of the application. In particular, Patient stores data about patients (*<P1>* and *<P2>* are examples of tuples); PerformedExamination stores the examinations performed by each patient (*<PE1, P1, Blood Tests, Liver function, 27/6/2016>* and *<PE2, P2, Urofluxometry, -, 28/7/2016>* are examples of tuples); and PerformedExaminationValue stores data about the filled parameters of the examination performed by a patient (examples of tuples are: *<PE1, Creatinine, 1.02>* and *<PE2, Volume, 157>*).

Figure 5 presents the application screen shown for the examination parameter Volume that is of type float and is measured in milliliters. The metadata stored in the database table Parameter, in particular, supplies the (meta)data to be shown in the forms presented in the front-end user interface. The clinical data that will be filled in by the application users is then stored in the database tables that store clinical data and is further available for future visualization and analysis.

3 Clinical Time Series Data Clustering and Visualization

A major advantage of medical information systems is the fact that they are capable of storing data from a large population of patients, hence providing a large source of data for discovering trends in disease progression that might be difficult to uncover by physicians in their daily clinical practice. In order to make such datasets useful, the medical information system needs to provide means to find groups of similar patients with basis on their personal and medical history and enable the exploration of the characteristics of the patients in these groups. We propose an approach to address this requirement, using an off-the-shelf state-of-the-art clustering algorithm for time series to find groups of patients with similar variation of relevant clinical parameters. We complemented the data analysis performed by the clustering algorithm with a visualization module to enable physicians to explore the results for each group of patients discovered by the algorithm. This approach can be applied to any medical parameter that varies over time.

3.1 Creating Time-Series Data Clusters

Clustering algorithms are widely used in biological research, among many other fields, for a variety of tasks [9]. The use of classical clustering techniques such as k-means has important disadvantages when used with biological or clinical data:

1. These algorithms tend to find clusters of similar size, hence they may not find interesting, relatively small clusters.

2. Each element (patient) is assigned to one and only one cluster, while it may display behavior similar to more than one (albeit at different intervals in time), or to none.
3. These algorithms compute clusters using all dimensions (which, in time series data, translate to time points) at once, hence they may not cluster together time series that are similar in all but one or two time points.

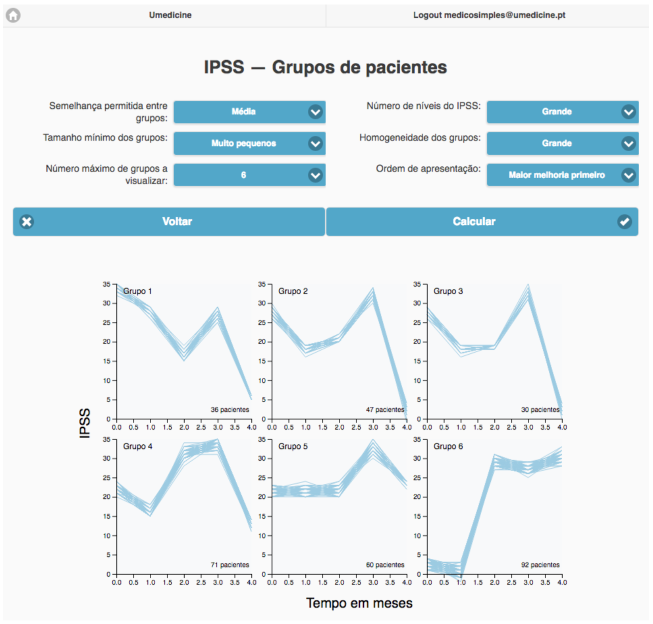
Biclustering algorithms are a family of clustering algorithms that overcome these disadvantages. Due to the similar size of genomic and medical datasets, we expect that biclustering's benefits apply to medical time series as well. Biclustering algorithms find groups of patients with basis on a subset of the time points instead of using all time points at once. In other words, they produce a local model instead of a global model [5]. The input data of Biclustering algorithms is represented as a data matrix, where each row represents the evolution of a given medical parameter for a patient, each column represents a time point and each matrix element holds a value for a medical parameter. Biclustering algorithms find clusters of rows in the input data matrix that are similar across a subset of the columns. The complexity of Biclustering algorithm depends on the applied criteria of similarity between rows, but most formulations are NP-hard [5, 7]. Hence, most biclustering algorithms resort to heuristics without guaranteeing optimal solutions, or have prohibitive running times. The problem becomes tractable, however, in cases where the measured medical parameter is discrete and the search for similar series is restricted to contiguous time points [5]. The Contiguous Column Coherent Biclustering (CCC-Biclustering) algorithm [5] overcomes these restrictions to find clusters in time series data in time linear in the size of the input data matrix. However, many medical parameters of interest are not discrete. In the original paper, authors are concerned with changes in gene expression, which are measured in a continuous positive scale with fixed zero minimum. Hence, the authors apply a discretization strategy to represent the variation of the gene expression in three levels: significant increase, significant decrease and no change of gene expression. This strategy reflects their goal of finding groups of genes with similar expression variation patterns. For other problems, care must be taken to choose an appropriate discretization approach as well.

Due to its performance, we decided to use the CCC-Biclustering algorithm in Umedicine. We illustrate the application of CCC-Biclustering in Umedicine with International Prostate Symptom Score (IPSS) time series data. IPSS is a score calculated from a standardized questionnaire given to certain Urology patients, such as patients that suffer from Benign Prostatic Hyperplasia (BPH). Physicians request that BPH patients fill in the questionnaire several times during the treatment, obtaining scores that reflect the patients' evolution over time time. IPSS has a uniform scale of integer values between 0 (best-case scenario for the patient) and 35 (worst-case scenario for the patient). It is, thus, discrete by itself. However, imposing such a fine discrete scale of 36 values would rarely cluster patients together: only when they had a sequence of exactly equal scores would the algorithm consider them part of the same cluster. For this reason, it makes sense to use a coarser discretized scale: for example, a scale with six discretization levels would enable for a difference of up to 6 points in IPSS while still being able to capture increasing or decreasing trends, as variations of the IPSS value would often result in a change to a different level.

The CCC-Biclustering algorithm also computes statistics for each cluster that it finds. One of the most important of these statistics is a p-value that reflects how similar the patients in the same group are among themselves. This p-value is calculated with basis on a hypothesis test in which the null hypothesis assumes that a cluster, with its size and patient data, was randomly generated. The lower the value of the p-value, the smaller is the probability of finding this group of patients under this null hypothesis. Hence, patients in groups with lower p-values can be expected to be more similar.

3.2 Visualization of Clinical Time-Series Data Clusters

We developed an interactive visualization mechanism, and incorporated into the Umedicine application, to enable the exploration of time-series data as analyzed by the CCC-Biclustering algorithm described in Section 3.1. The visualization is composed of a matrix of line charts (illustrated in Fig. 6), where each chart corresponds to a cluster of patients computed by the CCC-Biclustering algorithm. Each chart also indicates the number of patients in the corresponding group.



IPSS — Grupos de pacientes

Semelhança permitida entre grupos: Média
Número de níveis do IPSS: Grande
Tamanho mínimo dos grupos: Muito pequenos
Homogeneidade dos grupos: Grande
Número máximo de grupos a visualizar: 6
Ordem de apresentação: Maior melhoria primeiro

Grupo 1: 36 pacientes
Grupo 2: 47 pacientes
Grupo 3: 30 pacientes
Grupo 4: 71 pacientes
Grupo 5: 60 pacientes
Grupo 6: 92 pacientes

Fig. 6. Visualization of time series clustering in Umedicine for the IPSS parameter.

Users can set six different parameters to modify the visualization. Some of these parameters correspond to parameters of the CCC-Biclustering algorithm but were renamed for a more intuitive use by users with little knowledge of statistics. In the same order as in Fig. 6, they are:

1. **Allowed similarity between groups** (*Semelhança permitida entre grupos*): this parameter corresponds to the maximum overlap between clusters computed by the CCC-Biclustering algorithm. In other words, it defines the maximum percentage of patients in a group that can also be part of another group. The user can choose five levels of similarity: very small, small, intermediate, large and very large, corresponding to 1%, 5%, 10%, 25% and 50% maximum overlap.
2. **Number of IPSS levels** (*Número de níveis do IPSS*): corresponds to the number of intervals/symbols chosen to discretize the IPSS scale. The larger the number of IPSS levels, the narrower they are. Hence, a larger number of levels leads to clusters where patients are more similar to each other. The user has five options: very small, small, intermediate, large and very large, corresponding to 3, 6, 9, 12 and 18 levels of discretization.
3. **Minimum group size** (*Tamanho mínimo dos grupos*): the minimum number of patients in each cluster to be displayed. Charts with a number of patients inferior to the chosen number are not shown. The user has five options: very small, small, intermediate, large and very large. The 'very small' option sets the minimum number of patients per cluster/chart to two. The other options are computed as a function of the total number of patients in the dataset and correspond to 10%, 20%, 40% and 60% of the total number of patients, respectively.
4. **Group homogeneity** (*Homogeneidade dos grupos*): the larger this parameter, the smaller the maximum p-value of the clusters — as explained in Section 3.1; clusters whose p-values exceed this value are not shown. The user chooses among the options 'very small', 'small', 'intermediate', 'large' and 'very large', which correspond to p-values 0.25, 0.125, 0.083, 0.0625 and 0.05.
5. **Maximum number of clusters**/charts to show (*Número máximo de grupos a visualizar*): it takes into account the chosen presentation order of charts to filter out exceeding clusters.
6. **Presentation order** (*Ordem de apresentação*): users can choose between displaying charts ordered by decreasing number of patients per cluster, by decreasing or increasing overall change in IPSS. The overall change in IPSS here is the simple difference between IPSS at the final time point and the IPSS at the initial time point.

After setting these parameters as desired, the user generates the visualization by clicking/tapping the 'Calculate' (*Calcular*) button shown in Fig. 6).

Each chart can be selected with a mouse click or finger tap. This action overlays selected information about the corresponding group of patients, as illustrated in Fig. 7. The information shown includes the three most prevalent

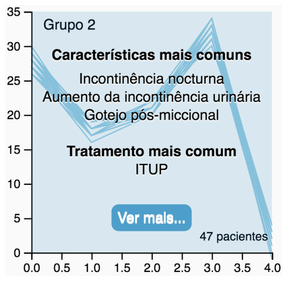
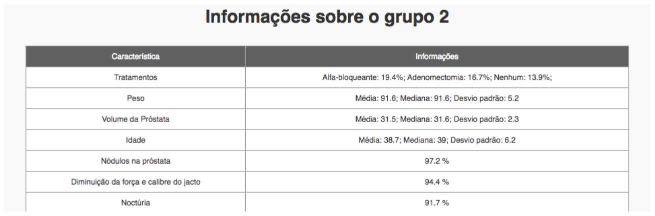


Fig. 7. Example of information displayed over a chart (from figure 6).

medical parameters in the patients belonging to the selected group (*Características mais comuns*) and the treatment given to most patients (*Tratamento mais comum*). Pressing the button at the bottom of the chart (*Ver mais...*) shows a table with detailed information about the selected group of patients (Fig. 8). This information includes the percentages of patients that were subject to the most common treatments for this group, statistical information about weight, prostate volume and age of these patients and, for each symptom and medical characteristic, the percentage of patients of the cluster that has this symptom or characteristic. This last part is sorted from highest to lowest.



Informações sobre o grupo 2

Característica	Informações
Tratamentos	Alfa-bloqueante: 19.4%; Adenomectomia: 16.7%; Nenhum: 13.9%;
Peso	Média: 91.6; Mediana: 91.6; Desvio padrão: 5.2
Volume da Próstata	Média: 31.5; Mediana: 31.6; Desvio padrão: 2.3
Idade	Média: 38.7; Mediana: 39; Desvio padrão: 6.2
Nódulos na próstata	97.2 %
Diminuição da força e calibre do jacto	94.4 %
Noctúria	91.7 %

Fig. 8. Example of information displayed when a user clicks the button 'Ver mais' (meaning 'get more information') on one of the charts shown in Fig. 7.

4 Experimental Validation

To validate our solution, we performed: i) performance tests to analyze the behavior of the clustering algorithm according to the number of patients and the number of data points over time; and ii) usability tests to evaluate the technique for visualizing and inspecting the clusters of patients produced by the clustering algorithm.

4.1 Performance Evaluation

To evaluate the performance of the clustering algorithm with medical time series data, we varied two input parameters: number of patients and number of time points per patient. We measured the time spent to analyze the data (clustering) and to display the results of the analysis (visualization). For the analysis (server side), we measured the time between receiving the request from the client up to the response from the server. Hence, it includes reading the data, computing the clusters, computing the related statistics and organizing data for presentation. Concerning the performance of displaying the results (client side), we measured the time between receiving the cluster data from the server and displaying the charts on the screen.

Experimental Setup To perform the evaluation of the clustering algorithm we had to generate synthetic data, because at this phase of the project we do not have real data from patients yet. We generated a synthetic dataset to be representative of a collection of patients that answered the IPSS questionnaire at several points in time after beginning a treatment. In addition to the IPSS scores we also generated data for various parameters relevant to the BPH pathology (e.g. weight, prostate volume).

We performed two experiments to test the CCC-Biclustering. In the first, we varied the number of patients for a fixed number of time points (5), and in the second, we varied the number of time points for a fixed number of patients (1,000). The fixed values for time points and patients were chosen to generate datasets with data close to reality. The experiment was performed on a 2009 MacBook Pro laptop computer with 8 gigabytes of 1066MHz DDR3 memory, a 2.66 GHz Intel Core 2 Duo processor and a SATA hard disk drive.

Results and Discussion The results obtained for both experiments are shown in Fig. 9. The left panel shows the behavior of our solution when we vary the number of patients and the right panel shows the behavior of our solution when we vary the number of time points per patient.

According to the CCC-Biclustering authors, the algorithm has a worst-case complexity linear in the size of all input parameters. Our results comply with this expectation when we vary the number of patients (Fig. 9-left), but not when we vary the number of time points (Fig. 9-right). We speculate that this deviation from linearity is caused by data pre- and/or post-processing steps, such as the computation of statistics. However, further investigation is necessary to explain this result.

Although the time to compute clusters for a large number of time points is relatively high, for realistic scenarios (less than 14 time points, corresponding to 14 appointments) the computation on the server side takes less than three minutes, making it possible to be used in an appointment scenario, where physicians can perform the analysis while attending the patient. The results only suffer a degradation of performance (with waiting times higher than 3 minutes) for more than 14 time points per patient, an unlikely scenario.

The performance measured for increasing number of patients is also encouraging: a matrix with 25,000 patients and the realistic column size of 5 time points

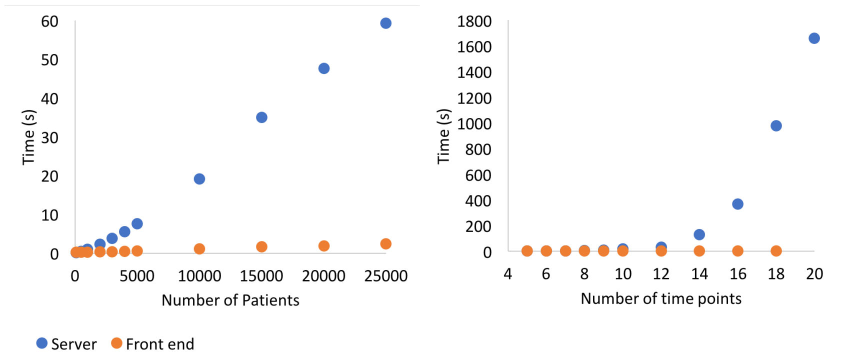


Fig. 9. Performance measurements for the CCC-Biclustering algorithm and the cluster visualization mechanism. Left panel: results for different numbers of patients for a fixed number of 5 time points per patient. Right panel: results for different numbers of time points per patient, for a fixed number of 1,000 patients.

could be analyzed in one minute. Assuming a linear trend, this means that data from hundreds of thousands of patients could be analyzed in a few minutes.

The processing of the visualization in the front-end, in turn, added a comparatively small waiting time, typically under half a second, which does not represent a performance concern.

4.2 Usability Tests

To complement the performance tests, we performed a usability test with users, to evaluate the visualization and inspection technique used to present the groups of patients and their main characteristics.

Experimental Setup We recruited ten volunteers to participate in the tests, being half of them male. Their ages were between 24 and 34 years old, and all of them had a university degree. The test consisted of using the application to perform six tasks (e.g identify the group of patients that improved their condition, the main characteristics of this group of patients, etc.). After completing the tasks, we asked users to answer a satisfaction questionnaire, composed of seven questions to rate the usage of our visualization mechanism and its characteristics (e.g. color scheme, presentation of information), and three open-answer questions inquiring about the greatest difficulties, best features and suggested modifications to the visualization.

Results and Discussion Table 1 summarizes information about the users and how they rated the visualization (from 1 to 5, 5 being the best) on several topics.

Overall, the results shown in Table 1 indicate that the users were pleased with their experience regarding the issues in questions Q1 to Q7. The lack of

Table 1. Usability test results

User	1	2	3	4	5	6	7	8	9	10	Average	Standard Error
Sex	F	M	F	M	F	M	M	M	F	F	NA	NA
Age	26	24	33	33	33	26	33	33	34	31	30.6	3.7
YUC	15	16	16	18	20	27	22	20	24	26	20.4	4.3
Q1	5	4	4	4	5	5	4	4	5	5	4.5	0.5
Q2	4	3	3	4	4	4	4	4	4	4	3.8	0.4
Q3	5	5	4	5	5	5	5	5	3	5	4.7	0.7
Q4	4	5	4	5	5	4	4	5	4	5	4.5	0.5
Q5	5	5	4	4	5	5	4	5	4	5	4.6	0.5
Q6	4	5	4	4	5	3	3	4	4	5	4.1	0.7
Q7	5	5	5	4	5	5	4	4	5	5	4.7	0.5

YUC: How long have you been using computers, in years?
Q1: The color scheme is appropriate to visualization the data
Q2: The options given to control patient group generation are easy to understand
Q3: The user can understand when a patient improves or gets worse
Q4: The user can understand that patients in the same group have similar IPSS along time
Q5: The user gets a good understanding of the characteristics of patients in the same group
Q6: It is easy to understand why patients are grouped together
Q7: The data displayed in the visualization table complement well the information shown in the corresponding chart

clarity of the meaning of the visualization parameters was the main complaint during the test. However, eight of the ten users recognized that once the meaning of the options was understood, they could use them and interpret how they impact the observed results. The experiment showed that the presentation of the options should be revised but, in general, users were able to complete the tasks successfully.

In the last part of the test, we asked users to describe what, in their opinion, were their main difficulties, the most positive aspects of the visualization and what they would like to see changed. Most users complained explicitly about the unintuitive options provided. Several users suggested to have the button that closes the table staying visible while the user scrolls up or down to inspect the table. Users tended to complimented the look and organization of the visualization page, with two of them mentioning the 'clean' look as a very positive aspect. Five users pointed the identification of common characteristics of the groups (Fig. 7) and the table with group statistics (Fig. 8) as the most positive aspects of the visualization. Three users indicated that they liked the interaction experience, especially its responsiveness. Other positive comments mentioned that it was easy to understand when patients improve or worsen.

5 Related Work

This section provides an overview of: (i) the most relevant commercial medical software applications that offer functionalities similar to the ones offered by Umedicine; and (ii) research works focusing on some of Umedicine's features.

There is a vast amount of medical software applications available in the market, providing support to clinical practice (in particular clinical appointments). The most important features offered are: (i) health center management aiming at managing appointment scheduling and payments; (ii) electronic prescription of medication; (iii) electronic health records to store clinical information about patients, such as symptoms and treatments; and (iv) population health management through graphics that show the evolution of health parameters for a set of patients. The first two features are out of the scope of the current paper, so we will not develop any further about them.

The software applications that provide features (iii) and (iv) and that are the most used in Portugal, are: Glintt HS[7], iMed[8] and MedicineOne[9]. Two additional software application used in Brazil (therefore, also supporting Portuguese) are IClinic[10] and HiDoctor[11]. Other software applications widely used according to the Medical Economics web site[12] are: CareAware[13], and eClinicalWorks[14], among others. We analyzed the functionalities provided by these tools based on the information available in the corresponding web sites and on the experience of our team's physician in his daily activities in Portuguese hospitals.

The result of the analysis performed is described in what follows. First, the forms and screens of these applications' user interface show all possible medical information (e.g., examinations or symptoms) independently of the medical specialty. This kind of user interface constitutes an overhead for the daily activity of physicians, because they waste time searching for the most adequate information field to fill in. Second, most of the applications typically support many free-form fields to be filled in by physicians. Consequently, unstructured (or textual) data is stored which prevents the application of data mining algorithms to discover knowledge from data for assisting physicians in decision taking. Third, none of the analyzed tools supports an electronic format of internationally accepted questionnaires for assessing patient's health status (for example, IPSS in Urology) and the corresponding answers given by patients. The electronic support of these questionnaires is important to allow patients to comfortably answer them (e.g. at home) and thus enabling the fast computation of a score that assesses

[7] http://www.glintt.com/
[8] https://www.imed.com.pt/
[9] http://www.medicineone.net/
[10] https://iclinic.com.br/
[11] https://hidoctor.com.br/
[12] http://medicaleconomics.modernmedicine.com/medical-economics/content/tags/top100ehrs/top-100-ehr-companies-part-1-4
[13] https://store.cerner.com/items/2
[14] https://www.eclinicalworks.com/

their health condition. Finally, very few of these tools (namely, careAware) supports the visualization of the evolution of a patient's health status over time. Moreover, a very small percentage of tools (e.g., eClinicalWorks) offers a functionality for analyzing the evolution of the health status of a set of patients (i.e., a population) over time.

In terms of research literature, we present two types of work: (i) those whose goal is to apply data mining algorithms to large medical datasets to extract useful information, and (ii) those whose goal is to apply visualization techniques to facilitate the exploration of data and analysis results by the end users. In the first group, we highlight the work by Donzé et al [3] that provides an example of how Logistic Regression can be used to make predictions about a patient outcome. Che et al [1] show how Bayes networks can be used to detect temporal patterns in time-series data, which is useful to predict the evolution of a patient over time. Finally, Najjar et al.'s paper [6] describes a powerful clustering approach that can be used to find groups of similar patients.

In the second group of research projects, LifeLines [8] was a seminal work developed for visualization of a patient's medical history. LifeLines2 [10,12,13] is a system designed to search and explore event sequences in temporal categorical data from multiple records of patient data. Gravi++ [4] is an application that clusters patients according to the similarity of their ordinal categorical attribute values. Furthermore, Gravi++ supports the visualization of the evolution of patients through time with an animation controllable by the user.

6 Conclusions

We described Umedicine, a software system to support clinical activity, by keeping medical information up to date and stored in an organized way. It provides easy, quick and always-on access to a large amount of clinical data. The medical information is persistently kept in an electronic and structured format. It provides secure authentication for four types of user, and a specific user interface for each. Additionally, the system includes a clustering algorithm suited to discover groups of patients with similar medical histories and a visualization mechanism to explore the clustering results, showing the main characteristics, treatments and statistics of the patient clusters.

Experimental evaluation revealed that users were able to understand the groups of patients created and the reasons (characteristics) why they were in the same group, and they were also able to identify the situations where patients improved or worsen their condition. Despite this, our solution still needs some improvements. One is to redesign the options of the visualization mechanism for an easier understanding by users. The clustering capabilities also have plenty of room for expansion: a variety of machine learning algorithms, such as Bayesian networks or artificial neural networks, among others, can be integrated in the system to suggest diagnosis, treatments or predict treatment outcomes based on the patient's history. Finally, it is important to test the application in a real clinical environment, with physicians as users and data from real patients. This experience will certainly guide future decisions regarding new functionalities to be integrated in the system and eventually determine the development lines that will be followed.

References

1. Che, Z., Kale, D., Li, W., Bahadori, M.T., Liu, Y.: Deep Computational Phenotyping. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 507–516. KDD '15, ACM, New York, NY, USA (2015)
2. Christensen, T., Grimsmo, A.: Instant availability of patient records, but diminished availability of patient information: A multi-method study of GP's use of electronic patient records. BMC Medical Informatics and Decision Making 8, 12 (2008)
3. Donzé, J., Aujesky, D., Williams, D., Schnipper, J.L.: Potentially Avoidable 30-Day Hospital Readmissions in Medical Patients: Derivation and Validation of a Prediction Model. JAMA Internal Medicine 173(8), 632 (Apr 2013)
4. Hinum, K., Miksch, S., Aigner, W., Ohmann, S., Popow, C., Pohl, M., Rester, M.: Gravi++: Interactive Information Visualization to Explore Highly Structured Temporal Data. J. UCS 11(11), 1792–1805 (2005)
5. Madeira, S.C., Teixeira, M.C., Sa-Correia, I., Oliveira, A.L.: Identification of Regulatory Modules in Time Series Gene Expression Data Using a Linear Time Biclustering Algorithm. IEEE/ACM Trans. Comput. Biol. Bioinformatics 7(1), 153–165 (Jan 2010)
6. Najjar, A., Gagné, C., Reinharz, D.: Two-Step Heterogeneous Finite Mixture Model Clustering for Mining Healthcare Databases. In: 2015 IEEE International Conference on Data Mining (ICDM). pp. 931–936 (Nov 2015)
7. Peeters, R.: The maximum edge biclique problem is NP-complete. Discrete Applied Mathematics 131(3), 651–654 (Sep 2003)
8. Plaisant, C., Mushlin, R., Snyder, A., Li, J., Heller, D., Shneiderman, B.: LifeLines: Using visualization to enhance navigation and analysis of patient records. Proceedings of the AMIA Symposium pp. 76–80 (1998)
9. Prelić, A., Bleuler, S., Zimmermann, P., Wille, A., Bühlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9), 1122–1129 (May 2006)
10. Rind, A.: Interactive Information Visualization to Explore and Query Electronic Health Records. Foundations and Trends® in Human–Computer Interaction 5(3), 207–298 (2013)
11. Smith, R.: What clinical information do doctors need? BMJ 313(7064), 1062–1068 (Oct 1996)
12. Wang, T.D., Plaisant, C., Quinn, A.J., Stanchak, R., Murphy, S., Shneiderman, B.: Aligning Temporal Data by Sentinel Events: Discovering Patterns in Electronic Health Records. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 457–466. CHI '08, ACM, New York, NY, USA (2008)
13. Wang, T.D., Plaisant, C., Shneiderman, B., Spring, N., Roseman, D., Marchand, G., Mukherjee, V., Smith, M.: Temporal summaries: Supporting temporal categorical searching, aggregation and comparison. IEEE transactions on visualization and computer graphics 15(6), 1049–1056 (2009 Nov-Dec)